Industry Paper: Surrogate Models for Testing Analog Designs under Limited Budget – a Bandgap Case Study

Roderick Bloem
Graz University of Technology
Austria

Alberto Larrauri
Graz University of Technology
Austria

Roland Lengfeldner
Infineon Technologies AG
Austria

Cristinel Mateis
AIT Austrian Institute of Technology
Austria

Dejan Ničković
AIT Austrian Institute of Technology
Austria

Björn Ziegler
Infineon Technologies AG
Austria

ABSTRACT

Testing analog integrated circuit (IC) designs is notoriously hard. Simulating tens of milliseconds from an accurate transistor level model of a complex analog design can take up to two weeks of computation. Therefore, the number of tests that can be executed during the late development stage of an analog IC can be very limited. We leverage the recent advancements in machine learning (ML) and propose two techniques, artificial neural networks (ANN) and Gaussian processes, to learn a surrogate model from an existing test suite. We then explore the surrogate model with Bayesian optimization to guide the generation of additional tests. We use an industrial bandgap case study to evaluate the two approaches and demonstrate the virtue of Bayesian optimization in efficiently generating complementary tests with constrained effort.

CCS CONCEPTS

• **Hardware → Best practices for EDA**; **Simulation and emulation**; *Safety critical systems.*

KEYWORDS

analog design, testing, surrogate model, machine learning

ACM Reference Format:
Roderick Bloem, Alberto Larrauri, Roland Lengfeldner, Cristinel Mateis, Dejan Ničković, and Björn Ziegler. 2018. Industry Paper: Surrogate Models for Testing Analog Designs under Limited Budget – a Bandgap Case Study. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Analog integrated circuit (IC) verification plays an important role in the development of modern safety critical systems. It is widely accepted that for complex mixed-signal IC products, verification accounts for 60%–70% of the total project development effort. Analog designs exhibit complex dynamics and hence, simulation-based testing remains the preferred verification method used in practice.

Simulation-based testing of analog IC is a challenging activity. First, the verification workflow involves tasks, such as test input generation, which are typically handled manually and require considerable human expertise. Second, simulating transistor-level analog IC designs can require tremendous computation effort – ten milliseconds of circuit's real time can result in hours, days or even weeks of simulation time, depending on the design's complexity and its level of accuracy. Design teams are working under pressure to deliver their product on time and have limited budget to spend on verification. Given the above challenges, it is imperative to smartly devise test inputs that will minimize the overall simulation effort, while maximizing the chance of covering all critical scenarios.

In this paper, we leverage recent advances in machine learning to improve the process of testing analog ICs under *constrained budget*. We explore two main questions: (I) How to use machine learning to infer from (possibly already available) simulation traces a *surrogate model* of the design, and (II) how to generate test inputs guided by machine learning methods that maximize the accuracy of the surrogate model with the minimum number of simulations.

We investigate three different methods to respond to these two questions. First, we randomly generate test inputs and use an artificial neural network (ANN) to create a surrogate model of the design. This is our baseline ML approach that can be effectively used in the presence of an existing test bench and simulation traces. We then investigate Gaussian processes (GPs) with *maximum entropy sampling* (MES) to learn the accurate surrogate model with a smaller number of test inputs and design simulations. In both cases, we then employ the particle swarm optimization (PSO) algorithm to find critical test scenarios in the design. We finally explore GPs and Bayesian Optimization (BO) to find efficiently critical test scenarios without attempting to achieve a necessarily good overall accuracy of the surrogate model.

The proposed methodology and its evaluation are driven by an industrial bandgap case study. A bandgap for accurate voltage and current reference is a versatile component used in many complex mixed-signal systems and hence a good representative of a typical analog IC. The evaluation highlights the strengths of the proposed methods and demonstrates the effectiveness of applying machine learning for testing analog ICs via learning surrogate models.

Related work. Deshmukh et al. [3] apply BO to CPS testing in order to minimize the number of required simulations when searching

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

for the global minimum of a function expressed as the robustness of a signal w.r.t. a given logical specification. Coupled with dimensionality reduction techniques, their approach proves to scale with the increase of the input space dimensionality.

Zhang et al. [10] propose an approximated GP-based method to infer a confidence measure for estimating the probability that an STL specification is satisfied by a black-box system. The approach starts from the test cases used by an arbitrary falsification framework which fails to find a counterexample not fulfilling the specification. These test cases are used to train a GP regression model which for a given input signal of the black-box system predicts the robustness of the STL formula w.r.t. the output signal of the black-box system. The surrogate GP model is used to construct the objective function of an optimization problem which aims at finding the local minima w.r.t. the 95% lower confidence bound of the robustness predictions around the best n candidates selected from N data points sampled from the black-box input space, where $n \ll N$. The n local minimum data points are used to construct a multivariate Gaussian distribution which, in turn, is used to infer the target confidence measure, calculated as the probability that the robustness values of all n data points are positive.

2 INDUSTRIAL CASE STUDY

The case study consists of an analog bandgap for accurate voltage and current reference provided by Infineon. The goal is to identify an appropriate circuit design for the bandgap which ensures both (i) the output voltage remains close to $1.132V$, within the specification interval $[\ 1.099V, 1.165V\ ]$, and (ii) the output current remains close to $20\mu A$ with a maximum tolerance of $\pm 1\mu A$. In addition to design parameters defined by Infineon's engineers, the bandgap outputs are sensitive to some input parameters which reflect its operation conditions. Among these, we selected the most important parameters according to domain knowledge. These are the parameters which induce a non-negligible variation of the bandgap outputs. Table 1 shows the selected influencing parameters; these are mainly supply voltages of the bandgap and the operation temperature.

Since implementing circuit designs is costly, it is not feasible to produce a testing device for each design until a particular design choice is physically proven to fulfil the specification. To overcome this issue, Infineon uses a functionally-performant HSPICE compatible simulator capable of reproducing the physical behavior of a given circuit design with high fidelity. Thus, in order to find an appropriate design fulfilling the specification, validation tests can be performed on the simulator. More specifically, the simulator is first configured with a given circuit design. Subsequently, tests corresponding to different valuations of the influencing parameters are run on the configured simulator. The output voltages and currents resulting from these simulations are checked against the specification. If the specification check passes for all valuations from the test suite, the current circuit design is selected for implementation and testing on a physical device before mass production. Otherwise, the engineers make a different circuit design choice and reconfigure the simulator with the new circuit design. Finally, they start a new iterative validation process with the reconfigured simulator and the same test suite. Although production-related costs can be avoided with this approach, such a simulator is computationally very intensive and the number of simulations which can be performed in reasonable time remains limited.

Name	Definition	min	typical	max	Comments
VDDPD	global supply	1.08V	1.28V	1.41V	
VDDA_EVR	local supply	2.20V	2.35V	2.80V	
VDDA_HPBG	local supply	2.20V	2.35V	2.80V	set equal to VDDA_EVR
VREF	local supply	0.579V	0.60V	0.621V	set to 0.6V
VSS	local supply		0V		set to 0V
Temp	temperature	-40°C	40°C	+175°C	

Table 1:

Infineon's simulator can be accessed in either batch mode or interactive mode. In batch mode, multiple tests can be queued into the simulator from the beginning. In interactive mode, only individual tests can be given to the simulator, each after all previous simulations have been completed. The average execution time in batch mode of one simulation is approximately one minute, while in interactive mode it is approximately two minutes. This increase is mainly due to network latency and fixed time costs per simulation launch.

3 METHODOLOGY

Since the methodology is similar for the verification of both voltage and current specifications, in the rest of this paper we will only focus on the output voltage.

We consider the configured simulator of the bandgap as a black-box function F_S, which maps each valuation of the parameters $(VDDPD, VDDA\_EVR, Temp)$ to an output voltage. Thus, $F_S : D \rightarrow \mathbb{R}$, where $D = [1.08, 1.41] \times [2.2, 2.8] \times [-40.0, 175.0]$. Our goal is to determine whether the output voltage remains withing the specification interval, that is, $F_S(x) \in [1.099, 1.165]$ for all $x \in D$. Evaluating F_S at a point x amounts to running an expensive simulation of the bandgap. Hence, we assume that sampling F_S is costly, and only a limited budget of output queries is available.

We apply three different testing methods to address the above specification verification problem and compare their effectiveness. In the first two, we learn a cheap-to-evaluate surrogate model f of F_S from examples of input/output pairs. In the first method this is done by training an artificial neural network (ANN), and in the second by fitting a Gaussian process (GP) using maximum entropy sampling (MES). We subsequently find estimates of the min/max values of f by employing an optimization solver with f as objective function. We conclude that the specification is fulfilled if those estimates fall within the specification interval. In the third approach we search for F_S's extreme values directly via Bayesian optimization (BO), a black-box optimization method. As for the baseline, we consider random testing. Here, we select a fixed number of independent uniform random locations on D and conclude that F_S is correct if the outputs of all corresponding simulations fall into the specification interval.

In order to evaluate the different methods efficiently, we generate a large initial dataset. We pick 9 values for $VDDPD$, 9 for $VDDA\_EVR$ and 12 for $Temp$, all (almost) equidistant in their domains and including the interval margins. All combinations of these values result in 972 points with good coverage of D. These are fed into simulator

in batch mode to obtain a set B of 972 samples. During each experiment, we randomly select 20% of B's samples to form a test set T, used for evaluation purposes. The remaining 80% of B is left to be used by the corresponding testing method as required.

We dedicate the rest of the section to detailing the testing methods under consideration. For this, we introduce the models of interest: artificial neural networks and Gaussian processes.

Artificial Neural Networks. ANNs [4] are machine learning models inspired by the networks of biological neurons. ANN are composed of artificial neurons. Formally, a neuron is a real-valued function $\phi(x_1, \ldots, x_n)$ with several real inputs. Each input connection i, $i = 1, \ldots, n$, is associated with a weight w_i which adjusts as learning proceeds. The neuron ϕ consists of the composition of two functions: (1) the weighted sum of its inputs, i.e. $z = w_1x_1 + \ldots + w_nx_n$, and (2) the activation function g. Hence, $\phi = g \circ z$. Neurons are typically aggregated into layers. An ANN is just a collection of layers connected together. The first layer of an ANN, which is the layer receiving external data, is called the input layer. The layer which produces the final result of the ANN is called the output layer. The intermediate layers between the input and output layers are called hidden layers. Training an ANN is the procedure of tuning the input weights of its neurons, i.e. the model parameters, until the ANN makes good predictions on unseen examples. Training is typically performed via the backpropagation algorithm introduced in [9].

We use an ANN architecture with three hidden layers, each consisting of 100 neurons with the relu activation function. The input layer consists of three neurons corresponding to the coordinates of D, and the output layer consists of a single neuron with the linear activation function. We use TensorFlow 2 [1] with Python 3 to train the ANN with the ADAM optimizer [6] and a learning rate of 0.001 for 1000 epochs.

Gaussian Processes. Let $m : D \to \mathbb{R}$ and let $k : D^2 \to \mathbb{R}$ be a covariance function (see [8]). A Gaussian process (GP) $f \sim \mathcal{GP}(m, k)$ is a collection of random variables $(f(x))_{x \in D}$ satisfying that (1) for any finite set $S \subset D$, the variables $(f(x))_{x \in S}$ follow a multivariate normal distribution, (2) $\mathrm{E}[f(x)] = m(x)$ for all $x \in D$, and (3) $Cov(f(x), f(y)) = k(x, y)$ for all $x, y \in D$. GPs can be used in regression tasks to model unknown functions. The mean $m(x)$ gives a predicted value for $f(x)$, while the standard deviation $\sigma(x) = \sqrt{k(x, x)}$ quantifies the uncertainty in that prediction. Importantly, given some set of observations $f(x_1) = y_1, \ldots, f(x_k) = y_k$, the random process can be "updated" via conditioning, using Bayes' rule. This results in a posterior stochastic process f' which is also distributed like a GP with new mean m' and covariance k'. In our experiments, we use GPs as statistical models for Infineon's voltage regulator F_S. For this, we employ the GP implementation available in Tensorflow 2.

The training of a GP $f \sim \mathcal{GP}(m, k)$ refers to the task of selecting mean m and covariance function k, as well as their parameters based on a given set of observations, called the training set. In our experiments, we normalize the input and output space around the training set, and pick the zero mean function $m \equiv 0$. For k, we employ the square exponential covariance function with differently weighted dimensions, given by $k(x, x') = \sigma_f^2 \exp\left(-1/2\|\Lambda(x - x')\|^2\right)$, where Λ is a diagonal matrix with elements $\lambda_1, \lambda_2, \lambda_3$. In order to choose the hyperparameters σ_f, $\lambda_1, \lambda_2, \lambda_3$ we follow the "log likelihood maximization" procedure described in [8]. We use the gradient-based optimizer ADAM for this task.

Now we are in conditions to describe the different testing methods employed during the experimental section.

3.1 ANN-Based Testing

Given a budget n of simulations, the algorithm selects n random samples from the set B to build the training set. This set is used to learn an ANN surrogate model f of the bandgap simulator F_S. Afterwards, we employ particle swarm optimization [5] (PSO) to find f's extreme values, as implemented in PySwarm library[1]. Finally, we compare those values against the specification.

3.2 Maximum Entropy Sampling

We use an adaptive method to fit a GP $f \sim \mathcal{GP}(m, k)$ to F_S. At each round t we call $O_t \subseteq D \times \mathbb{R}$ to the set of samples observed by the algorithm so far. We denote by $m_t : D \to \mathbb{R}$ and $\sigma_t : D \to \mathbb{R}$ the mean and the standard deviation of the posterior GP f_t, obtained by conditioning f to the observations O_t. The first set O_0 consists of 25 samples chosen at random from the batch simulations B. Hyperparameters of the GP are trained using those samples. During each round t, the algorithm selects a new location $x_t \in D$ to run an interactive simulation and add the corresponding sample to O_t, yielding O_{t+1}. The point x_t is selected by maximizing $\sigma_t(x)$ via PSO. This follows an approach from Bayesian experimental design where sample locations are chosen to maximize so-called entropy of a model [7]. When the budget n of samples is exhausted, i.e. $|O_t| = n$, the surrogate model f_t is used to estimate F_S's extreme values. This is done by running PSO on m_t, as with the ANN algorithm. Finally, the resulting estimates are checked against the specification.

3.3 Bayesian Optimization

Bayesian optimization [2] is a model-based technique for finding the global maximum of unknown objective functions. The algorithm is largely the same as MES, described above. The main difference lies in how the location x_t for the next interactive simulation is chosen at each round. During MES, x_t was a "maximally informative location". Now x_t is picked as a location where F_S is likely to produce a high value. This is done by maximizing an upper confidence bound (UCB) acquisition function $g_t : D \to \mathbb{R}$, given by $g_t(x) = m_t(x) + \beta_t^{1/2}\sigma_t(x)$, where $\beta_t = 4\ln(t + 1) + 10$. When the simulation budget runs out, the maximum sample value so far is used as an estimate for the maximum of F_S. To estimate the minimum of F_S, the optimization procedure is run on $-F_S$ instead. As in MES, the GP f_t could be used as a surrogate model of F_S, but f_t is only expected to be accurate around F_S's high values.

3.4 Empirical min/max values

We consider random testing as a baseline. Given a budget n, the algorithm runs n simulations at independent uniformly random data points in D. In this method, we use the min/max values among

[1] https://pythonhosted.org/pyswarm/

the n observed output voltage values as estimates for F_S's extreme values. We call these the empirical min/max values, respectively.

4 EXPERIMENTAL RESULTS

In this section, we detail the results of our experiments, related to the three methods ANN, BO, and MES. All experiments were run ten times and all colored regions in the graphs indicate the 95% confidence interval for the corresponding quantities. We address two questions: (1) How does the precision of the surrogate models obtained through each of the methods evolve with the number of samples, and (2) how do maxima/minima predicted according to the different methods vary with the number of samples?

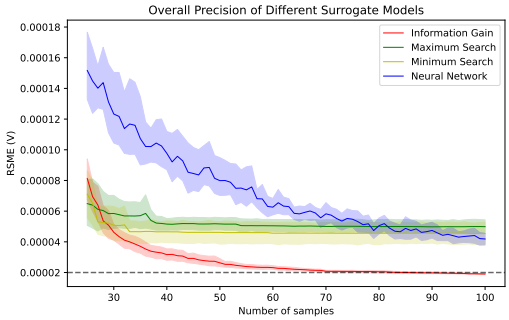

Figure 1: Learning curves comparison for surrogate model performances under maximum budget of 100 samples.

Figure1 relates to question (1), and plots how the test error of the different surrogate models evolve with the number of samples. It shows two versions of BO: one where the bandgap's value is being maximized, and other where it is minimized. The test error of a surrogate model $f : D \to \mathbb{R}$ is computed by taking the root mean square error (RMSE) on the test set T, defined as $\sqrt{m}$, where $m = \sum_{(x,y)\in T}(f(x) - y)^2/|T|$.

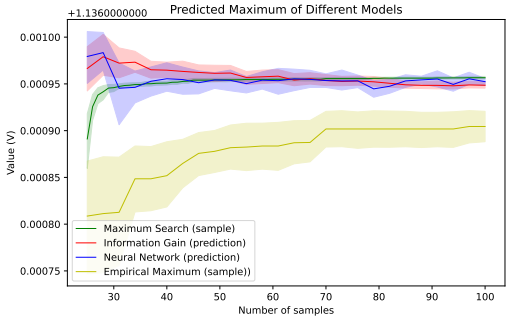

Figure 2: Learning curves comparison for the max search under maximum budget of 100 samples for all models.

As expected, MES and both versions of BO yield similarly precise models just after training on the initial set O_0 containing 25 samples. The initial training in these algorithms is performed the same way, so differences here are due to randomness in the experiments. Afterwards, the precision of the BO models plateaus in the range of $4\cdot10^{-5}$ - $6\cdot10^{-5}$ Volts of RMSE. In contrast, MES improves quickly in the beginning and stabilizes around $2\cdot10^{-5}$ Volts of RMSE at 80 samples. The ANN model starts off worse than the others, but catches up to BO within the first 100 samples. If a greater budget is allowed, the ANN model plateaus around $2\cdot10^{-5}$ Volts at 350 samples. Here, the MES approach seems clearly beneficial, even considering the fact that interactive simulations are twice more expensive than batch simulations. In any case, the RMSE achieved by all methods is orders of magnitude below the size of the specification interval for the bandgap, $[1.099V, 1.165V]$.

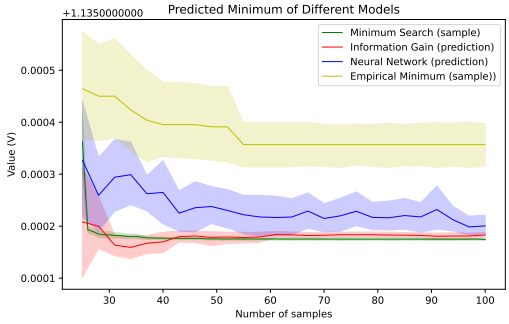

Figure 3: Learning curves comparison for the min search under maximum budget of 100 samples for all models.

Figures 2 and 3 address question (2). We consider four different methods now: ANN, BO, MES - as before- , and random sampling (RS). Figure 2 depicts how the predicted maxima given by the different methods evolve, and Figure 3 shows the minima. All ANN, BO and MES perform significantly better than random sampling. As shown in the graphs, BO achieves the most consistent results, both during the search of the maximum and that of the minimum. This is evidenced by the fact that BO's predictions exhibit the smallest standard error from the start. Those predictions seem to stabilize around $1.13695V$ at 50 samples, and around $1.1352V$ at 30 samples during the maximization and minimization tasks, respectively. The ANN and MES algorithms also stabilize quickly, in the range of 60 samples, achieving similar values to the ones obtained with BO, although the estimates given by ANN and MES appear to be slightly more conservative as the number of samples increases. The values obtained by all methods fall well within the specification interval of the bandgap, indicating high design robustness.

5 CONCLUSION

In this work we addressed the task of specification verification of a system under the constraint that the number of tests which can be performed on the system is limited. We described three different techniques and used an analog bandgap for accurate voltage and current reference as industrial case study to show that Bayesian optimization achieves the best results.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv:1012.2599 [cs]*, Dec 2010. arXiv: 1012.2599.

[3] Jyotirmoy Deshmukh, Marko Horvat, Xiaoqing Jin, Rupak Majumdar, and Vinayak S Prabhu. Testing cyber-physical systems through bayesian optimization. *ACM Transactions on Embedded Computing Systems (TECS)*, 16(5s):1–18, 2017.

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[5] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995.

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[7] Chun-Wa Ko, Jon Lee, and Maurice Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research*, 43(4):684–691, 1995.

[8] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006.

[9] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[10] Zhenya Zhang and Paolo Arcaini. Gaussian process-based confidence estimation for hybrid system falsification. In *International Symposium on Formal Methods*, pages 330–348. Springer, 2021.